

# **MEDICAL INVENTORY MANAGEMENT**

**College Name: BHARATHIDASAN COLLEGE OF ARTS AND SCIENCE**

**College Code: BRU3D**

**TEAM ID: NM2025TMID24629**

**TEAM MEMBERS:**

**Team Leader Name :**

**VISHWA S**

**EMAIL :**

[Vishwa12050@gmail.com](mailto:Vishwa12050@gmail.com)

TEAM MEMBER 1 : SAKTHIYANANTHAN V

EMAIL : Sakthiyananthan8@gmail.com

TEAM MEMBER 2 : SAKTHIKUMAR N

EMAIL : Sakthikumar739711@gmail.com

TEAM MEMBER 3 : VETTAIYAN P

EMAIL : Vettaianvettaiyan20@gmail.com

## **1. INTRODUCTION**

### **1.1 Project Overview**

This project is a comprehensive Salesforce application to streamline and manage various operational aspects of medical inventory. The system aims to efficiently maintain supplier details, manage purchase orders, track product details and transactions, and monitor the expiry dates of products. Maintain detailed records of suppliers, including contact information. Catalog product information, including descriptions, stock levels. Monitor and track product expiry dates to avoid using expired items. Comprehensive reports to performance, and purchase orders.


## **1.2 Purpose**

The Medical Inventory Management System is a comprehensive Salesforce application designed to streamline and manage various operational aspects of the medical inventory. It can efficiently maintain supplier details, manage purchase

# **DEVELOPMENT PHASE**

### **Creating Developer Account:**




By using this URL - <https://www.salesforce.com/form/developer-signup/?d=pb>



# Build enterprise-quality apps fast and get hands-on with Agentforce and Data Cloud.

Sign up for your Developer Edition.

- ✓ Build apps fast with drag-and-drop tools
- ✓ Go further with Apex code
- ✓ Build AI agents with Agentforce
- ✓ Harmonize your data with Data Cloud
- ✓ Ground Agentforce with structured and unstructured data
- ✓ Integrate with anything using APIs



## Sign up for your Developer Edition

A free Salesforce Platform environment with Agentforce and Data Cloud

First name

Last name

Job title

Work email

Company

Country/Region


India







Your org may be provisioned on or migrated to Hyperforce, Salesforce's public cloud infrastructure.


☐ I agree to the Main Services Agreement – Developer Services and Salesforce Program Agreement. I acknowledge, as described in the Developer Documentation: (1) the Developer Edition includes autonomous and other generative AI features; and (2) Salesforce may limit use of those features and the org, and may terminate any org that has been inactive for 45 days.

We value your privacy. To learn more, visit our [Privacy Statement](#).

<https://www.salesforce.com>




**Setup**

[Home](#)
[Object Manager](#)

[SETUP > OBJECT MANAGER](#)

**project**

Details
Fields & Relationships
Page Layouts
Lightning Record Pages
Buttons, Links, and Actions
**Compact Layouts**
Field Sets
Object Limits
Record Types
Related Lookup Filters
Search Layouts
List View Button Layout
Restriction Rules
Sharing Rules

**Compact Layouts**
1 Items, Sorted by Label


[New](#)
[Compact Layout Assignment](#)

LABEL	API NAME	PRIMARY	MODIFIED BY	LAST MODIFIED
System Default	SYSTEM	✓		








A Lightning App in Salesforce is a collection of items that work together to serve a particular function for the end-users. These items can include standard and custom objects, tabs, utilities, and other productivity tools. Lightning Apps are designed to provide a more intuitive and efficient user experience compared to traditional Salesforce apps

>>COMPACT LAYOUT

## >FIELDS & RELATIONSHIP



Search Setup



Setup

Home

Object Manager

SETUP > OBJECT MANAGER

project

Details

Fields & Relationships

Page Layouts

Lightning Record Pages

Buttons, Links, and Actions

Compact Layouts

Field Sets

Object Limits

Record Types

Related Lookup Filters

Search Layouts

List View Button Layout

Restriction Rules

Scoping Rules

**Fields & Relationships**

4 Items, Sorted by Field Label

Quick Find

New

Deleted Fields

Field Dependencies

Set History Tracking

FIELD LABEL	FIELD NAME	DATA TYPE	CONTROLLING FIELD	INDEXED
Created By	CreatedById	Lookup(User)		
Last Modified By	LastModifiedById	Lookup(User)		
Owner	OwnerId	Lookup(User,Group)		✓
product ID	Name	Text(80)		✓

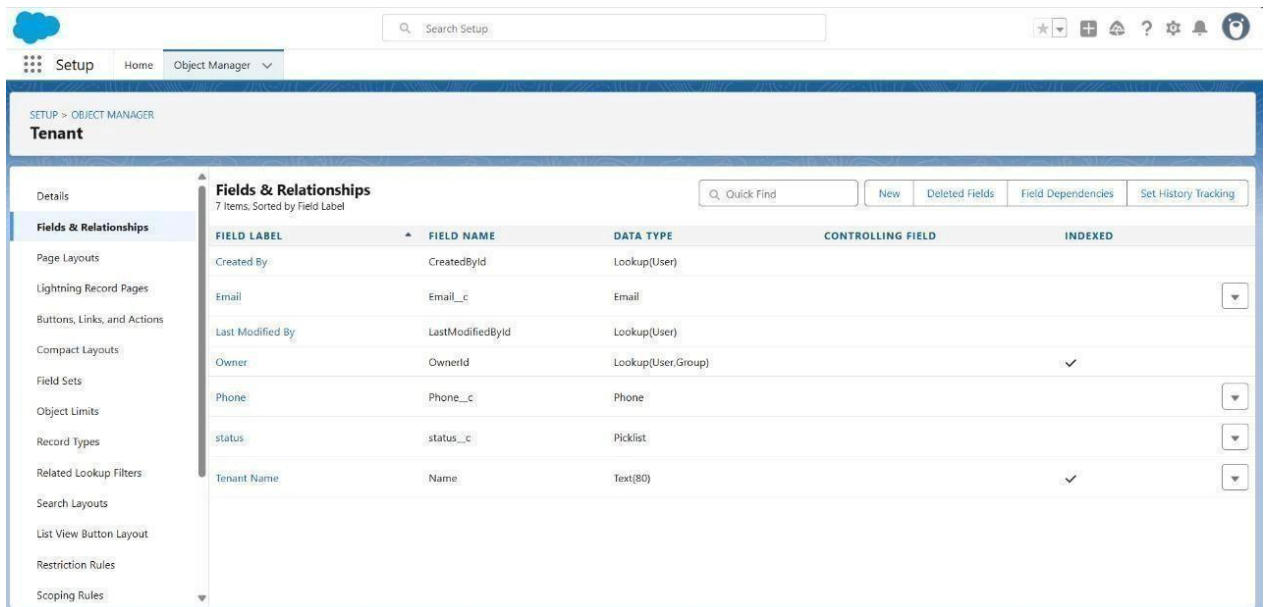
Object	Field Name	Data Type
<b>Product</b>	Product ID(Standard)	Text
	Product Name	Text
	Product Description	Text Area
	Minimum Stock Level	Number(18, 0)
	Current Stock Level	Number(18, 0)
	Unit Price	Currency(16, 2)
	Expiry Date	Date
<b>Purchase Order</b>	Purchase Order ID(Standard)	Text
	Supplier ID	Lookup(Supplier)
	Order Date	Date
	Expected Delivery Date	Date
	Actual Delivery Date	Date
	Order Count	Roll-Up Summary (COUNT Order Item)
	Total Order Cost	Currency(16, 2)
<b>Order Item</b>	Order Item ID(Standard)	Text
	Product ID	Lookup(Product)
	Purchase Order ID	Master-Detail(Purchase Order)
	Quantity Ordered	Number(18, 0)
	Quantity Received	Number(18, 0)
	Unit Price	Formula(Currency)
	Amount	Formula(Currency)

<b>Inventory Transaction</b>	Transaction ID(Standard)	Text
	Purchase Order ID	Lookup(Purchase Order)
<b>Supplier</b>	Transaction Date	Date
	Transaction Type	Picklist
	Total Order Cost	Formula(Currency)
	Supplier ID(Standard)	Text
	Supplier Name	Text
	Contact Person	Text
	Phone Number	Phone
	Email	Email
	Address	TextArea

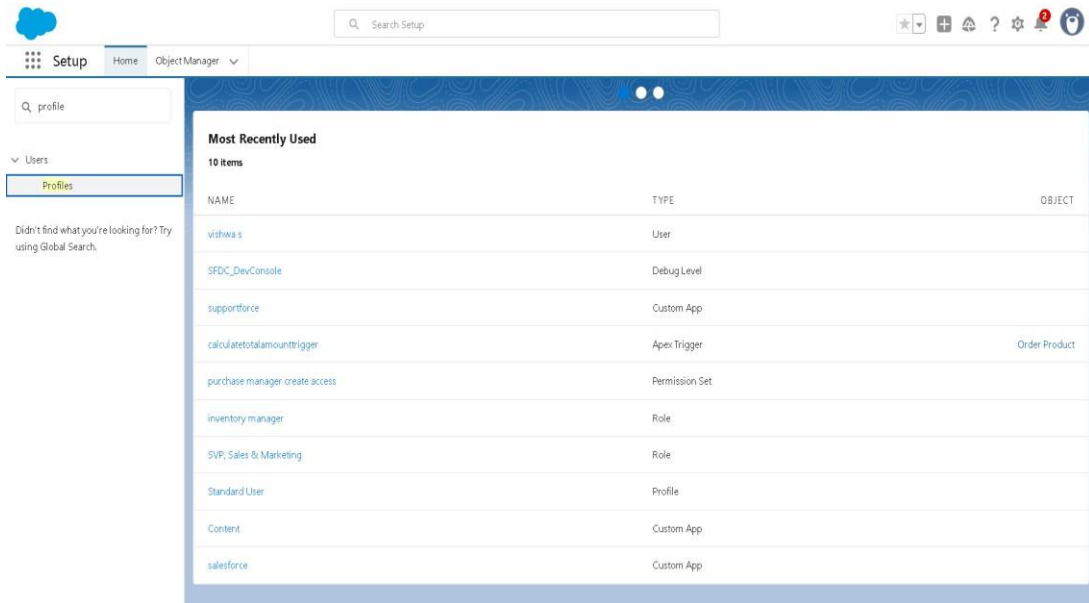
# Activity 1: Creating a Text Field in Product Object

To create fields in an object:

1. Click the gear icon and select Setup. This launches Setup in a new tab.
2. Click the Object Manager tab next to Home.
3. Select Product custom object.
4. Select Fields & Relationships from the left navigation
5. Click on New
6. Select Text field, click Next
7. Enter Field Label as “Product Name” and Length 255.
8. Select Required Field.
9. Click Next, Next, then Save & New.



- Developed Profile App with relevant tabs



## Activity 1: To create an Inventory Manager Profile

1. Go to setup >> type profiles in quick find box >> click on profiles >> clone the desired profile (Standard User) >> enter profile name (Inventory Manager) >> Save.

While still on the profile page, then click Edit.

2. Select the Custom App settings as default for the Medical Inventory Management.
3. Scroll down to Custom Object Permissions and Give access permissions as mentioned in the below diagram.

Custom Object Permissions						
	Basic Access				Data Administration	
	Read	Create	Edit	Delete	View All	Modify All
Inventory Transactions	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Order Items	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Products	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Purchase Orders	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Suppliers	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

- 4.
- 5.
6. Change the password policies as mentioned :
7. User passwords expire in should be " never expires ".
8. Minimum password length should be " 8 ", and click save.



## Activity 2: To create an Purchase Manager Profile

1. Go to setup >> type profiles in quick find box >> click on profiles >> clone the desired profile (Standard User) >> enter profile name (Purchase Manager) >> Save.
2. While still on the profile page, then click Edit.
3. Select the Custom App settings as default for the Medical Inventory Management.

SETUP Profiles

Set the permissions and page layouts for this profile.

Profile Edit

Name: Purchase Manager

User License: Salesforce

Description:

Custom Profile: ☒

Custom App Settings

	Visible	Default
All Tabs (standard__AllTabSet)	<input checked="" type="checkbox"/>	<input type="radio"/>
Analytics Studio (standard__Insights)	<input checked="" type="checkbox"/>	<input type="radio"/>
App Launcher (standard__AppLauncher)	<input checked="" type="checkbox"/>	<input type="radio"/>
Bolt Solutions (standard__LightningBolt)	<input checked="" type="checkbox"/>	<input type="radio"/>
Community (standard__Community)	<input checked="" type="checkbox"/>	<input type="radio"/>
Content (standard__Content)	<input checked="" type="checkbox"/>	<input type="radio"/>
Data Manager (standard__DataManager)	<input checked="" type="checkbox"/>	<input type="radio"/>
Digital Experiences (standard__SalesforceCMF)	<input checked="" type="checkbox"/>	<input type="radio"/>
Lightning Usage App (standard__LightningUsageApp)	<input checked="" type="checkbox"/>	<input type="radio"/>
Marketing CRM Classic (standard__Marketing)	<input checked="" type="checkbox"/>	<input type="radio"/>
Medical Inventory Management (Medical_Inventory_Management)	<input type="checkbox"/>	<input checked="" type="radio"/>
Queue Management (standard__QueueManagement)	<input checked="" type="checkbox"/>	<input type="radio"/>
Sales (standard__LightningSales)	<input checked="" type="checkbox"/>	<input type="radio"/>
Sales (standard__Sales)	<input checked="" type="checkbox"/>	<input type="radio"/>
Sales Console (standard__LightningSalesConsole)	<input checked="" type="checkbox"/>	<input type="radio"/>
Salesforce Chatter (standard__Chatter)	<input checked="" type="checkbox"/>	<input type="radio"/>
Salesforce Scheduler Setup (standard__LightningScheduler)	<input type="checkbox"/>	<input type="radio"/>
Sample Console (standard__ServiceConsole)	<input type="checkbox"/>	<input type="radio"/>
Service (standard__Service)	<input checked="" type="checkbox"/>	<input type="radio"/>
Service Console (standard__LightningService)	<input checked="" type="checkbox"/>	<input type="radio"/>
Site.com (standard__Sites)	<input checked="" type="checkbox"/>	<input type="radio"/>
Subscription Management (standard__RevenueCloudConsole)	<input checked="" type="checkbox"/>	<input type="radio"/>
WDC (standard__Work)	<input checked="" type="checkbox"/>	<input type="radio"/>

4. Scroll down to Custom Object Permissions and Give access permissions as mentioned in the below diagram.

Custom Object Permissions

	Basic Access				Data Administration	
	Read	Create	Edit	Delete	View All	Modify All
Inventory Transactions	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Order Items	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Products	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Purchase Orders	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Suppliers	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

5. Change the password policies as mentioned :
6. User passwords expire in should be " never expires ".
7. Minimum password length should be " 8 ", and click save.

Password Policies

User passwords expire in: Never expires

Enforce password history: 3 passwords remembered

Minimum password length: 8

Password complexity requirement: Must include alpha and numeric characters

Password question requirement: Cannot contain password

Maximum invalid login attempts: 10

Lockout effective period: 15 minutes

Obscure secret answer for password resets: ☐

Require a minimum 1 day password lifetime: ☐

Don't immediately expire links in forgot password emails: ☐

Save Save & New Cancel

- implemented Flows for monthly rent and payment success
- To create a validation rule to a Lease Object

# FUNCTIONAL AND PERFORMANCE TESTING

## Performance Testing

- Trigger validation by entering duplicate tenant-property records

The screenshot displays the Salesforce Setup interface, specifically the 'Roles' page. The left sidebar shows the navigation menu with 'Setup' selected. The main content area is titled 'Understanding Roles' and includes a description: 'Set up your Role Hierarchy to control how your organization reports on and accesses data.' Below this, there is a section for 'Sample Role Hierarchy' with a dropdown menu set to 'Territory-based Sample'. A hierarchical diagram illustrates the structure: 'Executive Staff' (CEO, President, CFO, VP, Sales) at the top, followed by 'Western Sales Director', 'Eastern Sales Director', and 'International Sales Director' in the middle, and 'Western Sales Rep', 'Eastern Sales Rep', and 'International Sales Rep' at the bottom. Each role box includes a brief description of its responsibilities and access permissions. For example, the 'Executive Staff' role allows viewing and editing data, rolling up forecasts, and generating reports for all users below, but cannot access data of other Executive Staff. The 'Sales Rep' roles allow viewing and editing data, rolling up forecasts, and generating reports for all users directly below, but cannot access data of users above or at the same level. At the bottom right, there is a 'Set Up Roles' button and a checkbox labeled 'Don't show this page again'.

- Validation Rule checking

Roles in Salesforce are used to control record-level access and define the hierarchy of an organization, determining the level of visibility and sharing of records among users. Roles work in conjunction with profiles to provide a robust security model. While profiles control what actions users can perform (object and field permissions), roles control which records users can see based on their position in the hierarchy.

# Activity 1 : Create a Purchasing Manager Role.

1. Go to quick find >> Search for Roles >> click on Set Up Roles.



2. Click on Expand All and click on add role under SVP, Sales & Marketing role.
3. Give Label as “Purchasing Manager” and Role name gets auto populated. Then click on Save.



Role Edit

## New Role

**Role Edit**

Label	<input type="text" value="Purchasing Manager"/>
Role Name	<input type="text" value="Purchasing_Manager"/> ⓘ
This role reports to	<input type="text" value="SVP, Sales &amp; Marketing"/> 🔍
Role Name as displayed on reports	<input type="text"/>

# PERMISSION SETS

Setup

Home

Object Manager

permission

Users

Permission Set Groups

Permission Sets

Custom Code

Custom Permissions

Didn't find what you're looking for? Try using Global Search.

SETUP

Permission Sets

Help for this Page

Permission Sets

On this page you can create, view, and manage permission sets.

All Permission Sets | Edit | Delete | Create New View

New

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z Other All

Action	Permission Set Name	Description	License
<input type="checkbox"/> Clone	(Legacy) Data Cloud Data Aware Specialist	This Data Cloud permission set will be deprecated in Spring '24. Learn ...	Customer Data Platform
<input type="checkbox"/> Clone	(Legacy) Data Cloud Marketing Admin	Allows access to Data Cloud Setup if the user is also a Salesforce adm...	Customer Data Cloud for Marketing
<input type="checkbox"/> Clone	(Legacy) Data Cloud Marketing Manager	This Data Cloud permission set will be deprecated in Spring '24. Learn ...	Customer Data Platform
<input type="checkbox"/> Clone	(Legacy) Data Cloud Marketing Specialist	This Data Cloud permission set will be deprecated in Spring '24. Learn ...	Customer Data Platform
<input type="checkbox"/> Clone	(Legacy) Data Cloud for Marketing Data Aware Specialist	This Data Cloud permission set will be deprecated in Spring '24. Learn ...	Customer Data Cloud for Marketing
<input type="checkbox"/> Clone	(Legacy) Data Cloud for Marketing Manager	This Data Cloud permission set will be deprecated in Spring '24. Learn ...	Customer Data Cloud for Marketing
<input type="checkbox"/> Clone	(Legacy) Data Cloud for Marketing Specialist	This Data Cloud permission set will be deprecated in Spring '24. Learn ...	Customer Data Cloud for Marketing
<input type="checkbox"/> Clone	Access Agentforce Default Agent	Gives users access to the default Agentforce agent in Salesforce.	Agentforce (Default)
<input type="checkbox"/> Clone	Agent Platform Builder	Allow access to agent platform.	Agent platform builder
<input type="checkbox"/> Clone	Agentforce Default Admin	Allows users to build and manage in-org copilots.	Agentforce (Default)
<input type="checkbox"/> Clone	Agentforce Service Agent Configuration	Build and manage autonomous AI service agents.	Agentforce Service Agent Builder
<input type="checkbox"/> Clone	Agentforce Service Agent Object Access	Access knowledge articles and manage cases and contacts as an auto...	Agentforce Service Agent User
<input type="checkbox"/> Clone	Agentforce Service Agent Secure Base	Set up and use Agentforce Service Agent actions with enhanced data s...	Agentforce Service Agent User
<input type="checkbox"/> Clone	Agentforce Service Agent User	Analyze topics and perform actions as an autonomous AI service agent.	Agentforce Service Agent User

1-25 of 85 | 0 Selected | Previous Next | Page 1 of 4

## • Test flows on payment update

File • Edit • Debug • Test • Workspace • Help • < >

calculatetotalamounttrigger.apxt \* A

Code Coverage: None | API Version: 64 | Go To

```
1 trigger CalculateTotalAmountTrigger on Order_Item__c (after insert, after update, after delete, after undelete) {
2
3     // Call the handler class to handle the logic
4
5     CalculateTotalAmountHandler.calculateTotal(trigger.new, trigger.old, trigger.isInsert, trigger.isUpdate, trigger.isDelete, trigger.isUn
6
7 }
8 public class CalculateTotalAmountHandler {
9
10
11
12     // Method to calculate the total amount for Purchase Orders based on related Order Items
13
14     public static void calculateTotal(List<Order_Item__c> newItems, List<Order_Item__c> oldItems, Boolean isInsert, Boolean isUpdate, Boole
15
16
17
18     // Collect Purchase Order IDs affected by changes in Order_Item__c records
19
20
```

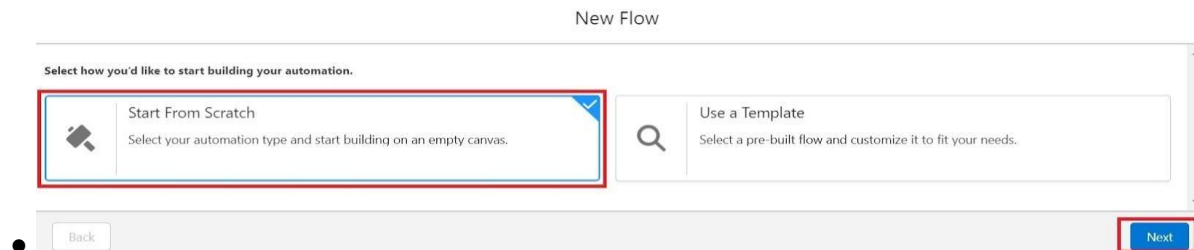
Logs Tests Checkpoints Query Editor View State Progress Problems

User	Application	Operation	Time	Status	Read	See
------	-------------	-----------	------	--------	------	-----

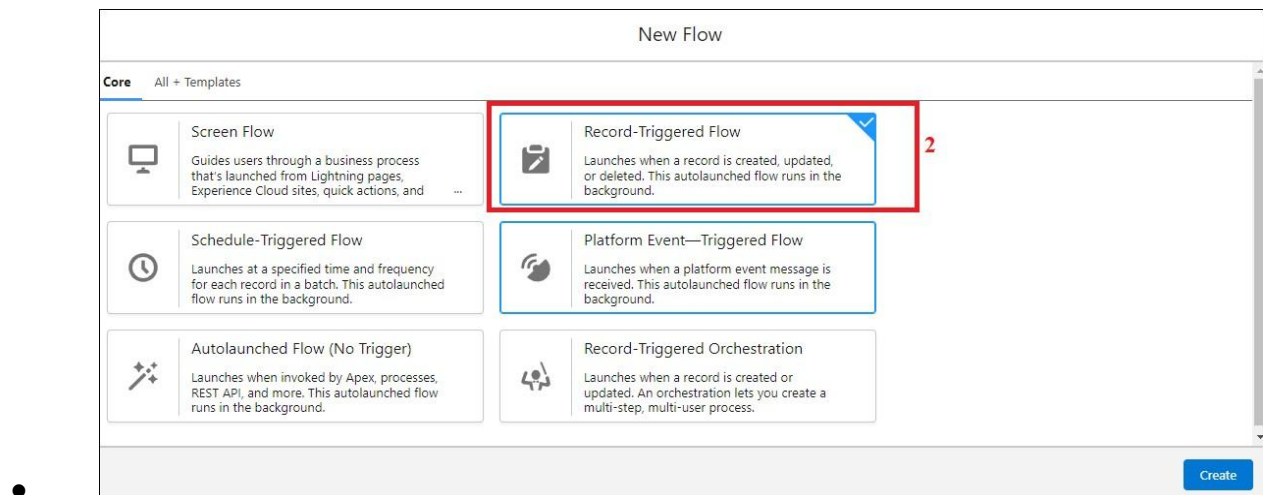
Filter Click here to filter the log list

- Approval process validated through email alerts and status updates
- **Activity 1 : Create Flow to update the Actual Delivery Date.**

2. Go to setup >> type Flow in quick find box >> Click on the Flow and Select the New Flow >> Start From Scratch .



3. Select the record Triggered flow. Click on create.



4. Under Object select “Purchase Order”

5. Select A record is created or updated

**Configure Start**

**Select Object**

Select the object whose records trigger the flow when they're created, updated, or deleted.

\* Object

Purchase Order **3**

**Configure Trigger**

\* Trigger the Flow When:

☐ A record is created  
☐ A record is updated  
☒ A record is created or updated **4**  
☐ A record is deleted

## 6. Set Entry Conditions : None

## 7. Select Fast Field Updates and click on Done

**Set Entry Conditions**

Specify entry conditions to reduce the number of records that trigger the flow and the number of times the flow is executed. Minimizing unnecessary flow executions helps to conserve your org's resources.

If you create a flow that's triggered when a record is updated, we recommend first defining entry conditions. Then select the **Only when a record is updated to meet the condition requirements** option for When to Run the Flow for Updated Records.

Condition Requirements

None **5**

\* Optimize the Flow for:

**Fast Field Updates**

Update fields on the record that triggers the flow to run. This high-performance flow runs *before* the record is saved to the database.

**Actions and Related Records**

Update any record and perform actions, like send an email. This more flexible flow runs *after* the record is saved to the database.

☐ Include a Run Asynchronously path to access an external system after the original transaction for the triggering record is successfully committed

## 8. Under the record trigger flow click on the "+" icon and select Get Records.

## 9. Enter Label as " Get Purchase Record ".

## 10. For Object select Purchase Order.

## 11. For Condition Requirements , select All Conditions are Met(AND)

- For the first condition select as follows:
- Field: Id
- Operator: Equals
- Value: {!\$Record.Id}

**Get Records**

\* Label: Get Purchase Record **8**

\* API Name: Get\_Purchase\_Record

Description:

Get Records of This Object

\* Object: Purchase Order **9**

Filter Purchase Order Records **10**

Condition Requirements: All Conditions Are Met (AND)

Field	Operator	Value
Id	Equals	\$Record > Record ID

+ Add Condition

- 12.** For How many Records to store Select Only the First Record. **13.** For How to Store Record Data select Choose fields and let Salesforce do the rest. Select Field: Order\_Date\_\_c. Click on Done.

**How Many Records to Store**

☒ Only the first record

☐ All records

**How to Store Record Data**

☐ Automatically store all fields

☒ Choose fields and let Salesforce do the rest

☐ Choose fields and assign variables (advanced)

Select Purchase Order Fields to Store in Variable

Field

ID

Field

Order\_Date\_\_c

+ Add Field

- 14.** In the Flow Builder, click on the Manager tab on the left-hand side >> Click on New Resource >> In the Resource Type dropdown, select Variable.
- 15.** Enter API name as ActualDeliveryDate >> Select Data type as Date >> Click on Done.
- 16.** From the Toolbox drag and drop Assignment element.



17. Enter the label as "Assignment".

18. Set Variable Values:

- a) Variable : {!ActualDeliveryDate}
- Operator : Equals
- Value : {!\$Record.Order\_Date\_\_c}
- b) Variable : {!ActualDeliveryDate}
- Operator : Add • Value : 3
- 
- 

**Assignment**

\* Label: Assignment

\* API Name: Assignment\_1

Description

**Set Variable Values**

Each variable is modified by the operator and value combination.

Variable	Operator	Value
ActualDeliveryDate	Equals	\$Record > Order Date
ActualDeliveryDate	Add	3

+ Add Assignment

19. Click Done

20. From the Toolbox drag and drop Update Records element and connect to the Assignment element.

21. Enter the label as "Updating Purchasing Order".

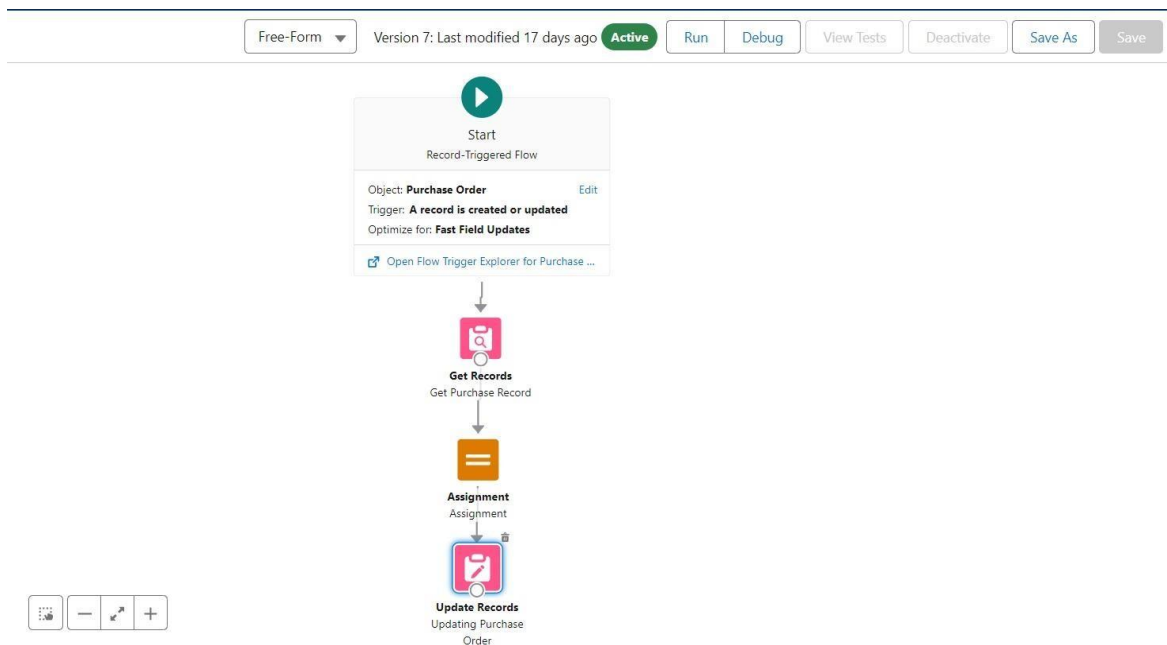
22. How to Find Records to Update and Set Their Values : Use the Purchase Order record that triggered the flow

23. Set Filter Conditions : None -Always Update Record

24. Set Field Values for the Trip Record as

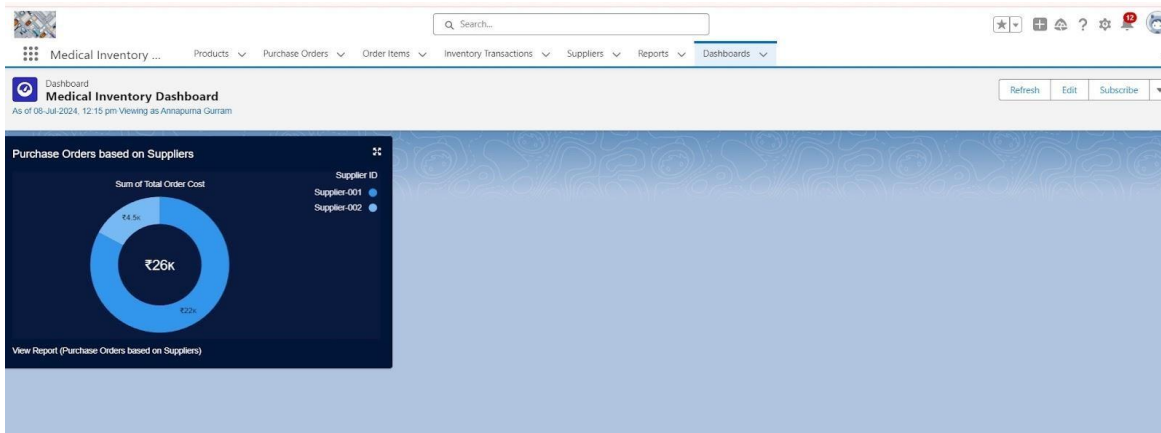
- Field : Actual\_Delivery\_Date\_\_c
- Value : {!ActualDeliveryDate}

25. Click Done
26. Save the flow as “Actual Delivery Date Updating”.
27. Activate the flow.



# RESULTS

## Output Screenshots



- Tabs for Property, Tenant, Lease, Payment
- Email alerts
- Request for approve the leave
- Leave approved
- Leave rejected

- Flow runs

- Trigger error messages

---

## **ADVANTAGES & DISADVANTAGES**

---

# CONCLUSION

The Lease Management System successfully streamlines the operations of leasing through a structured, automated Salesforce application. It improves efficiency, communication, and data accuracy for both admins and tenants.

---

## APPENDIX

- **Source Code:** Provided in Apex Classes and Triggers **Test.apxt:** trigger CalculateTotalAmountTrigger

on Order\_Item\_\_c (after insert, after update, after delete, after undelete) {

// Call the handler class to handle the logic

CalculateTotalAmountHandler.calculateTotal(trigger.new, trigger.old, trigger.isInsert, trigger.isUpdate, trigger.isDelete, trigger.isUndelete);

}

Step 4:

i) In the Developer Console window, go to the top menu and click on "File".

ii) Select New: From the dropdown menu under "File", select "New".

iii) Choose Apex Class: Name it as CalculateTotalAmountHandler

```
public class CalculateTotalAmountHandler {
```

```
    // Method to calculate the total amount for Purchase Orders based on related Order Items    public static void calculateTotal(List<Order_Item__c> newItem, List<Order_Item__c> oldItem, Boolean isInsert, Boolean isUpdate, Boolean isDelete, Boolean isUndelete) {
```

```
        // Collect Purchase Order IDs affected by changes in Order_Item__c records        Set<Id> parentIds = new Set<Id>();
```

```
        // For insert, update, and undelete scenarios        if (isInsert || isUpdate || isUndelete) {            for (Order_Item__c ordItem : newItem) {                parentIds.add(ordItem.Purchase_Order_Id__c);            }        }
```

```
        // For update and delete scenarios        if (isUpdate || isDelete) {            for (Order_Item__c ordItem : oldItem) {                parentIds.add(ordItem.Purchase_Order_Id__c);            }        }
```

```
        // Calculate the total amounts for affected Purchase Orders        Map<Id, Decimal> purchaseToUpdateMap = new Map<Id, Decimal>();
```

```

if (!parentIds.isEmpty()) {
    // Perform an aggregate query to sum the Amount__c for each Purchase Order
    List<AggregateResult> aggrList = [
        SELECT Purchase_Order_Id__c, SUM(Amount__c) totalAmount
        FROM Order_Item__c
        WHERE Purchase_Order_Id__c IN :parentIds
        GROUP BY Purchase_Order_Id__c
    ];

    // Map the result to Purchase Order IDs      for
    (AggregateResult aggr : aggrList) {
        Id purchaseOrderId = (Id)aggr.get('Purchase_Order_Id__c');      Decimal
        totalAmount          =          (Decimal)aggr.get('totalAmount');
        purchaseToUpdateMap.put(purchaseOrderId, totalAmount);
    }

    // Prepare Purchase Order records for update
    List<Purchase_Order__c> purchaseToUpdate = new List<Purchase_Order__c>();      for (Id
    purchaseOrderId : purchaseToUpdateMap.keySet()) {
        Purchase_Order__c purchaseOrder = new Purchase_Order__c(Id = purchaseOrderId,
        Total_Order_cost__c = purchaseToUpdateMap.get(purchaseOrderId));      purchaseToUpdate.add(purchaseOrder);
    }

    // Update Purchase Orders if there are any changes      if
    (!purchaseToUpdate.isEmpty()) {
        update purchaseToUpdate;
    }
}
}
}
}

```

Save it.